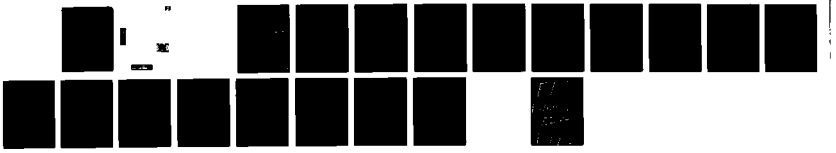
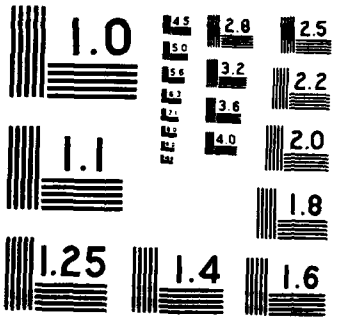
AD-A206 863 THE STATE-OF-THE-ART IN EXTRACTING EIGENVALUES AND 1/1
EIGENVECTORS IN STRUCT..(U) CALIFORNIA UNIV BERKELEY
CENTER FOR PURE AND APPLIED MATHEMAT.. B N PARLETT
UNCLASSIFIED APR 87 PAM-373 N00014-85-K-0180 F/G 12/2 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3

THE STATE-OF-THE-ART IN EXTRACTING EIGENVALUES AND EIGENVECTORS IN STRUCTURAL MECHANICS

Beresford N. Parlett

Department of Mathematics
University of California, Berkeley

ABSTRACT

There have been significant advances in the art of extracting eigenvalues and eigenvectors in the last 15 years. This fact is not widely appreciated.

Topics addressed: the special problems arising in Mechanics, reduction to standard form, subspace iteration, Lanczos algorithms, singular mass matrices.

Keywords: Symmetric matrices, iteration, (KR)

The contents of this report will be a chapter in the " **State-of-the-art Surveys on Computational Mechanics** *" to be published by the American Society of Mechanical Engineers and edited by* **Prof. A. K. Noor**.

The author gratefully acknowledges support from ONR Contract N00014-85-K-0180. He thanks Mr. Jian Le for his expert word processing.

NOMENCLATURE

We apologize for not using familiar engineering notational practices but there is a good reason. Few special notations are needed in this paper and so a simple system (avoiding brackets, braces, and underlines) suffices.

whole numbers	$j, ..., n$ (Fortran convention)
real numbers	$\alpha, \beta, \cdots$
column vectors	$a, b, \cdots$ (except $i, ..., n$, and t for time)
components of vectors	$a(1), a(2), \cdots$
matrices	$A, B, \cdots$
identity matrix	$I = (e_1, ..., e_n)$
imaginary unit	i (not j)
transpose	a^t or A^t
time derivative	$\dot{u}$
norm	$\|v\| := \sqrt{v^t v} = \sqrt{v(1)^2 + \cdots + v(n)^2}$

1. INTRODUCTION

In the previous State-of-the-Art survey there was no chapter devoted exclusively to eigenvalue extraction. There is little point in casting back to 1950 and we shall confine our attention to the last decade.

The primary achievement of the 1970's was the incorporation of robust and fairly efficient eigenvalue solvers into general finite element packages such as NASTRAN, SAP, ADINA, ... Consequently the dynamic analyses of structures with about 10^3 degrees of freedom (d.o.f.) became routine activities all over the engineering world. Systems with 10^4 d.o.f. were analyzed and even 10^5 d.o.f. was not a pipe dream. See [8], [10], [20], [21], [22], [28], [38] for example.

These programs permitted more intensive analyses of aircraft designs and so helped in the production of the recent fuel-efficient jetliners, (e.g. Boeing 767).

In order to set the stage for the rest of the paper some facts of life must be appreciated. First, the arithmetic effort required to compute all the eigenvalues of an $n \times n$ symmetric matrix is less than the effort required to form the product of two such matrices. The only requirement on n is that it be possible to hold two full $n \times n$ arrays in the fast memory of the computer. These

matrices are said to be small for the given computer system. See [1], [2], and [61].

The standard eigenvalue problem for small matrices is solved whether the matrix is symmetric or not. The methods are utterly reliable and fast, see section **2**. Fortran programs are available in most computer centers. They can be found in libraries called EISPACK, IMSL, and NAG, among others. Even if an engineer wants only 3 eigenvalues (or eigenpairs) it is usually best to find them all when the matrix is small. The largest computed eigenvalue will be accurate to almost working precision (i.e. 13 correct decimals if the unit roundoff is 10^{-14}) and the smaller eigenvalues have the same absolute accuracy as the largest one. See [1]. Programs for extracting eigenvalues of small matrices should be used like the built-in subroutines for cosine or square root. There is no need for the engineer to write his own version. For small matrices it is all right to reduce the general linear eigenvalue problem to standard form explicitly, using matrix multiplication. See Reduction II (with $\sigma = 0$) in section **5**.

For large matrices the situation is different. Nevertheless it is still not advisable for an engineer to consult a book and try to implement methods described there.

During the 1980's there was gradual acceptance of the fact that the successful Subspace Iteration method, on which the 1970 eigensolvers were often based, was not the last word in efficiency. See [5], [41]. Some versions of the Lanczos algorithm promised to be an order of magnitude more efficient. For example, a block Lanczos eigensolver, produced by Boeing Computer Services under the leadship of L. Komzsik, was incorporated into the McNeal-Schwendler NASTRAN package during the summer of 1985. This sophisticated code does not require that any of the vectors it computes remain in the fast store (i.e. virtual memory has been accepted). This means that the program can analyze quite large structures on fairly small computers (large minis, such as the VAX 750). See [26], [38].

All these eigensolvers can be vectorized to some extent but they do not really exploit the full power of machines such as the CYBER 205 or the CRAY 2. See [57]. The situation may well have been rectified by the times these words appear in print.

In the remaining parts of this paper we will try to convey " the big picture " of the state-of-the-art in modal analysis and buckling. We shall point the reader to the open literature for more details and justification. Inevitably some valuable work will have been omitted from the references. Slighted researchers are urged to contact the author and fill the gaps in his knowledge.

2. TECHNIQUES FOR SMALL MATRICES

We give a brief outline of the methods used for small problems. If A is symmetric it is reduced to tridiagonal form T (i.e. $t(j, k) = 0$ if $| j - k | > 1$) by explicit orthogonal similarity transformations. If eigenvectors are wanted the transformation are accumulated and the cost of this aspect dominates all the others. The matrix T is reduced to diagonal form Λ by the QR algorithm. Fewer than n^3 multiplications are required to obtain all n eigenvalues. If all the eigenvectors are wanted the cost rises to 5 n^3.

For unsymmetric matrices the pattern is similar. The matrix B is reduced to Hessenberg form H ($h(j, k) = 0$ if $j > k+1$) by explicit orthogonal similarity transformations. Then H is reduced to block triangular form by the QR algorithm. The blocks on the diagonal are either 2×2, for each complex conjugate pair, or 1×1, for each real eigenvalue. The cost for all eigenvalues is about 5 n^3 multiplications, for eigenvectors too the cost rises to 15 n^3.

The general linear eigenvalue problem $A - \lambda B$ may be solved without inverting A or B by use of the QZ algorithm. This reduces A to block upper triangular form $\tilde{A}$ and B to upper triangular form $\tilde{B}$. The diagonal blocks of $\tilde{A}$ are either 2×2 or 1×1. The cost is approximately 15 n^3 multiplications. The best reference for all these algorithms, and more, is the EISPACK guide, see [1,2]. The reference for learning about the computation of eigenvalues is Wilkinson's classic tome " The Algebraic Eigenvalue Problem " (Oxford Univ. Press, 1966). It should be stressed that the understanding engendered by that book was translated into action in the "

Handbook for Automatic Computation : vol. II, Linear Algebra ", (Springer-Verlag, 1971) and the EISPACK, NAG, and IMSL program libraries are all based on it.

3. PROBLEM DESCRIPTION

A good place to start is the equation of motion of some structure. It may be written

$$M\ddot{u} + C\dot{u} + Ku = f(t),\ t>0,$$

where the vector $u = u(t)$ represents the state of the system and M,C, K are square matrices. The modes are those solutions of the homogeneous equation (i.e. $f \equiv 0$) with the special form $u(t) = exp(\ i\omega t\ )x$ ($i^2 = -1$), where x is a fixed vector. Thus the natural frequency ω, and x, the mode shape vector, must satisfy

$$(\ K + i\omega C - \omega^2 M\ )x = 0.$$

When $C = 0$, each eigenpair (ω, x) represents one of the free modes of vibration of a conservative model of the structure.

The algebraic eigenvalue equation given above is quadratic in ω but, when $C = 0$, is linear in ω^2. The computation of ω and x is more or less difficult depending on the structure of K, C, and M. We list some important cases in increasing order of difficulty. See [7], [14], [33], [34].

a) $C = 0$, K and M symmetric, positive definite (= s.p.d.).

b) $C = 0$, K s.p.d., M symmetric, positive semi-definite, and singular.

c) K, C, M all s.p.d. (dissipative system due to friction).

d) C skew-symmetric, K and M s.p.d. (rotating systems).

e) $C = 0$, K s.p.d., M sym. but indefinite (for buckling analysis, where $\lambda = \omega^2$ is not necessarily real). See [76].

f) C symmetric, indefinite, K not symmetric, M s.p.d. (rotating, dissipative structures, $\lambda = \omega^2$ need not be real). See [66].

Although cases c, d, e, f are being solved it is fair to say that they are still research topics as far as eigenvalue solvers are concerned. See [16], [28].

Key Features

Common to all the cases listed above are the following special requirements:

i) Only a few pairs (ω, x) are wanted, rarely more than 100. For vibration analysis, where ω is always positive, there are two standard demands; either all frequencies in a given interval or the m fundamental modes where $1 \leq m < 100$.

ii) Most of the elements in K, C, and M are zero. When finite elements are used to construct the matrices then they have a loosely banded appearance, depending strongly on the connections within the structure. Also the number of degrees of freedom keeps growing. In the 1980's 10^3 is typical but 10^5 is to be expected.

4. GENERAL COMMENTS

The special character of these tasks, embodied in i) and ii) above, turns our attention away from traditional methods that seek to diagonalize a symmetric matrix by means of a sequence of similarity transformations. See [79]. Most similarity transformations spoil the sparsity pattern in a matrix and are not cost effective when only 20 eigenvalues out of, say, 2000 are wanted.

The preferred methods today are all **sampling techniques.** They create a matrix (or better, a linear operator) and apply it to a sequence of carefully constructed vectors. From these

transformed vectors the dominant eigenvectors and their eigenvalues can be approximated.

The Inertia Count

However there is one quite different technique that is often used to check on the computed results. It is an application of Sylvester's Inertia Theorem. If K and M are symmetric, τ is any real number, and if Gaussian elimination, without interchanges, is performed on $K - \tau M$ to produce the triangular factorization

$$K - \tau M = LDL^t$$

where D is a diagonal matrix containing the pivots and L is lower triangular with ones on the diagonal and the multipliers elsewhere, then

" the number of eigenvalues of the pair (K, M) that are less than τ is just the number of negative pivots in D " See [4, p.46]

If any pivot is tiny it is best to change τ by say .001% and do the factorization again.

This useful, though expensive, facility is sometimes called a Sturm Sequence count for odd historical reasons. It is used to check that no eigenvalues have been missed by the algorithm. See [6], [70].

The arrival of vector computers at one end of the range and mini- and micro- computers at the other has made it virtually impossible to give simple cost comparisons between methods. See [75]. Nevertheless it often happens that the application of the operator (or matrix) to a vector dominates all other costs. So it is not unreasonable to compare the number of matrix-vector products required for the computation by various algorithms. Data transfer is also important.

When K and M have narrow bandwidth b the cost of the triangular factorization mentioned above is quite low ($b^2\, n$ ops) and consequently each step of inverse iteration is also cheap. If only three or four eigenpairs are wanted it is quite attractive to find them one at a time using combinations of zero finding techniques. The determinant search method of Bathe is a good example of this approach. See [7] for more details.

5. REDUCTION TO STANDARD FORM

The various problems described above each involve at least 2 matrices. Yet the most powerful techniques require a single matrix or operator. As will be explained below it is clearer to use the word operator than matrix. There are three ways to reduce the problem

$$(K - \omega^2 M)q = 0 \qquad (1)$$

to standard form

$$(B - \lambda I)x = 0 \qquad (2)$$

Let $M = \tilde{L}\tilde{L}^t$ be the Cholesky factorization of M. Here $\tilde{L}$ is a lower triangular matrix. Sometimes $\tilde{L}$ is called the square root of M but this is a bad abuse of language because the square root of M is defined to be the symmetric positive definite matrix X that satisfies $X^2 = M$.

I. Premultiply (1) by $\tilde{L}^{-t}$ and seek $\tilde{L}q$ in place of q to find

$$(\tilde{L}^{-t} K \tilde{L}^{-1} - \omega^2 I)\tilde{L}q = 0 \qquad (3)$$

Thus the operator B ($= \tilde{L}^{-t} K \tilde{L}^{-1}$) is symmetric. Of course M must be positive definite so the application of (I) is limited. There is no need to form the matrix B explicitly. See [43].

Method (I) is a poor one simply because we want eigenvalues ω_1^2, ω_2^2, ... close to 0 and these are much more expensive to compute than those near ∞.

II. Take any convenient value σ near the wanted eigenvalues. The value $\sigma = 0$ is a common but not very effective choice. Rewrite (1) as follows

$$(K - \sigma M - (\omega^2 - \sigma)M)q = 0,$$

$$((\omega^2 - \sigma)^{-1} I - (K - \sigma M)^{-1} M)q = 0,$$

$$((\omega^2 - \sigma)^{-1} I - \tilde{L}^t (K - \sigma M)^{-1} \tilde{L})\tilde{L}^t q = 0. \qquad (4)$$

Thus

$$B = \tilde{L}^t (K - \sigma M)^{-1} \tilde{L}.$$

Again there is no need to form B explicitly. The product $x = Bv$ is evaluated in 3 steps: $w \leftarrow \tilde{L}v$; solve $(K - \sigma M)u = w$ for u; $x \leftarrow \tilde{L}^t u$.

Any effective method for solving $(K - \sigma M)u = w$ may be used, direct or iterative. If the size of K and M is not too great then direct methods are popular. This requires a preliminary factorization,

$$K - \sigma M = LDL^t$$

as described in section 4. See [19], [60].

Note that B is symmetric and its eigenvalues $\lambda_1, \lambda_2, \ldots$, are related to the wanted frequencies by

$$\lambda_k = \frac{1}{(\omega_k^2 - \sigma)}, \quad k = 1, 2, \cdots$$

It is some of the largest eigenvalues of B that are wanted and these are the easiest to compute. Note also that M need only be positive semi-definite, $\tilde{L}$ can have some 0 elements on its diagonal.

The considerable cost of factoring $K - \sigma M$ is handsomely offset by the rapid convergence of the iterative methods to be described later.

III. It is not necessary that B be a symmetric matrix. It is sufficient that it be similar to a symmetric matrix. Consequently the last step in deriving (4) may be omitted. Thus

$$((\omega^2 - \sigma)^{-1} I - (K - \sigma M)^{-1} M)q = 0,$$

and

$$B = (K - \sigma M)^{-1} M.$$

The eigenvectors are not changed at all.

There is a price to pay for this convenience. At one or more places in the algorithms it is necessary to multiply a vector by M whereas use of reduction II avoids this step. See [60].

Both II and III are satisfactory, with a slight preference for III because it is simpler. It will be shown in a later section that the algorithms can be written so that there is no need to know whether or not M is singular or ill-conditioned. See [48] and [68].

The important point is that the user supplies a subroutine that returns Bv for any given vector v. This embodies the linear operator that sends v into Bv. Yet the matrix B is never formed explicitly. The goal now is to call that subroutine with a sequence of vectors v and calculate from the output the dominant few eigenvectors of B.

6. SUBSPACE ITERATION (called SI)

In Europe this is called the method of Simultaneous Iterations. See [12] and [31]. Fortunately it has the same acronym SI.

Programs based on this method are in widespread use throughout the engineering community for eigenvalue extraction. See [7], [8], [11], [12], [31]. The effectiveness of the programs comes from sophisticated implementations whose details cannot be described here. In other words there is still some art (or judgement) required to make the implementations robust. One of the cleverest implementations is H. Rutishauser's RITZIT algorithm presented in [79] but actually written in 1968-1969. For a brief account of it see [59, Chap.14] and [65]. RITZIT had little influence on engineers in the U.S.A. who, in the course of time, rediscovered some of Rutishauser's devices.

The basic idea is very simple. It is a block power method. The power method takes a starting vector v and keeps applying the operators B to produce the so-called Krylov sequence (or power sequence)

$$v,\ Bv,\ B(Bv),\ B^3v,\ B^4v,\ \cdots .$$

For almost all v the vector B^kv points in the direction of the dominant eigenvector of B as $k \to \infty$.

In a block version, with block size m, one starts with m starting vectors, the columns of a $n \times m$ matrix $V^{(0)}$. It is best to start with an orthonormal set, i.e.

$$(V^{(0)})^t V^{(0)} = I_m$$

The trouble with forming $B^kV^{(0)}$ for large k is that all its columns are dominated by the dominant eigenvector. This defect is easily cured by orthonormalizing the current set of m vectors from time to time. If this is done every time B is applied then one obtains one version of SI. However there is more than one way to orthonormalize a set of vectors although the Gram-Schmidt process is the most popular. In the present context there is a better way: use the Rayleigh-Ritz approximations.

Let V be any $n \times m$ matrix whose columns are linearly independent. There is a pretty recipe for obtaining the best set of approximate eigenvectors using just linear combinations of V's columns. We present it with implementation III of section 5 in mind.

The Rayleigh-Ritz Approximations from V

1. Form the $m \times m$ projection matrices (often called interaction matrices)
$$C := V^t(BV),\ E := V^t(MV)$$
2. Solve the full $m \times m$ generalized eigenvalue problem $(C - \theta_i E)g_i = 0$, i = 1, ..., m to find $G := (g_1, ..., g_m)$ and $\Theta := \text{diag} (\theta_1, ..., \theta_m)$. Normalize the g_i so that $g_i^t E g_i = 1$, i = 1, ..., m.
3. Form $W = VG$.

The vectors $w_1, ..., w_m$ are the desired approximate eigenvectors. In general some of the w_i will be much better approximations than others. The numbers $\theta_1, ..., \theta_m$ are the Rayleigh-Ritz approximate eigenvalues.

The essential SI algorithm is to keep repeating these Rayleigh-Ritz approximations.

Basic SI Algorithm

1. pick a full rank $V^{(0)}$

2. for k = 1, 2, ..., until satisfied
   a) obtain Rayleigh-Ritz approximations ($g_1^{(k)}$, $\theta_1^{(k)}$), ..., ($g_m^{(k)}$, $\theta_m^{(k)}$) from $V^{(k-1)}$
   b) compare $\theta_j^{(k)}$ with $\theta_j^{(k-1)}$ to monitor convergence, j = 1, ..., m
   c) if not satisfied set

$$v_j^{(k)} := V^{(k-1)} g_j^{(k)} \quad \text{for } each \ bad \ j,$$

$$v_j^{(k)} := v_j^{(k-1)}, \quad \text{if } \theta_j \ has \ converged.$$

For a fuller discussion of SI with attention to more details of implementation see:

K. J. Bathe and coworkers [7, Chap. 12]; [8] for his latest.

A. Jennings and coworkers [33, Chap. 10], and, in turn, [11], [12], [31], [34].

Both groups have an engineering orientation. The numerical analysts were mentioned at the beginning of section 6.

Important ingredients to successful usage of SI are

1. Choice of m
2. Choice of $V^{(0)}$
3. Mechanisms for avoiding convergence to already known eigenvectors.

Good variations of SI have been incorporated into a number of FEM packages.

There is only one fundamental criticism to be made of SI: it may use considerably more calls to the operator subroutine than are necessary. The explanation is very simple. At every step in SI the approximations $V^{(k-1)}$ are overwritten with better ones in $V^{(k)}$. This overwriting discards information and the penalty can be significant. In fact it is not necessary to increase storage costs to achieve a substantial saving in computational effort.

7. THE LANCZOS APPROACH

The power of the Lanczos algorithm comes from a fortunate approximation property of Krylov subspaces. Although it is quite surprising it is not difficult to state.

It is well known that the power sequence (see the third paragraph of section 6 for the power sequence), started from a random vector, converges almost always to the dominant eigendirection of the operator B. In principle a large number of steps might be necessary. Suppose we ask a different question. After $k-1$ steps of the power method how many eigenvectors of B could be approximated to 5 decimal accuracy by taking suitable linear combinations of our k vectors

$$v, Bv, B^2v, ..., B^{k-1}v \quad ?$$

Remember that 5 decimal accuracy in an approximate eigenvector y means 10 decimal accuracy in the approximate eigenvalue $\theta = (y^tBy)/(y^tMy)$.

Now for $k = 1$ the answer is none and for $k = n$, the number of degrees of freedom, the answer is n. However these extremes are not relevant. The amazing observation is that for values of k between 20 and $n/5$ the answer is about $k/2$. As k increases the hit ratio improves but for most applications in Mechanics it is about 50% or better. Consequently it should be possible to computer 25 eigenpairs with only 50 calls on the operator (i.e. 50 matrix-vector products). Yet after 50 steps the power method might well have produced only a 3 decimal approximation to a single eigenvector.

When one recalls that the starting vector is chosen at random this phenomenon is remarkable. The eigenvalues computed in this way will be those closest to the number σ that occurs in

the definition

$$B = (K - \sigma M)^{-1} M.$$

The phenomenon does not depend on n, the number of d.o.f.

The same thing occurs with the block power method. Suitable combinations of all the vectors in V, BV, B^2V, ..., B^9V (i.e. $9m$ matrix-vector products) will usually produce $3m$ accurate eigenvectors. The block approximation is slightly weaker than the single vector case which would give 9m/2 eigenvectors.

The difficulty in trying to exploit the phenomenon is that the power sequence is a hopeless one to work with in practice; it is nearly linearly dependent. The Lanczos algorithm computes a different set of vectors that nevertheless has the same approximation property as the power sequence. What is even better is that it is not necessary to keep all k Lanczos vectors in the fast memory.

The original attraction of the Lanczos algorithm came from the so-called three term recurrence relation presented as far back as 1950 by Lanczos in [39]. It can be described as follows.

Theoretically the Lanczos vectors are what you get by applying the Gram-Schmidt procedure, using the M-inner product, to the power vectors but that is not how they are computed. Call them q_1, q_2, ..., q_k. Then $q_1 := v \,/\, \|v\|_M$. Let us consider the construction of q_{k+1} from q_1, q_2, ..., q_k.

As in the power method B is applied to the last term in the current sequence to give $\tilde{r} := Bq_k$. Now we seek the part of $\tilde{r}$ that is M-orthogonal to q_1, q_2, ..., q_k. There is a beautiful result that shows that $\tilde{r}$ is already M-orthogonal to q_1, ..., q_{k-2} (in exact arithmetic). So it is only necessary to execute

$$\tilde{r} := Bq_k,$$
$$r := \tilde{r} - q_{k-1}\gamma_k - q_k\alpha_k$$

where

$$\gamma_k := \tilde{r}^t M q_{k-1},$$
$$\alpha_k := \tilde{r}^t M q_k = (\tilde{r} - q_{k-1}\gamma_k)^t M q_k$$

Then normalize r as follows

$$\beta_{k+1} := \| r \|_M,$$
$$q_{k+1} := r / \beta_{k+1}.$$

Unravelling this computation reveals that

$$Bq_k = q_{k-1}\gamma_k + q_k\alpha_k + q_{k+1}\beta_{k+1}$$

and this is the three term recurrence relation. A little more theory shows that, in exact arithmetic,

$$\gamma_k = \beta_k$$

so that there is no need to compute the γ_k. Theoretically the old Lanczos vectors q_1, ..., q_{k-2} can be sent off to secondary storage since they are not needed again until the approximate eigenvectors are computed at the end.

The coefficients α_i, β_i go to build up a tridiagonal matrix T_k as indicated,

$$T_k = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \cdot & \cdot & \\ & & \cdot & \cdot & \beta_k \\ & & & \beta_k & \alpha_k \end{bmatrix}$$

This matrix holds all the essential information. Notice how little work there is in computing q_{k+1} compared with one step of subspace iteration (SI).

Suppose k is the last step. To complete the calculation we proceed as follows,

1) Compute all eigenvalues and eigenvectors of T_k, $T_k s_i = s_i \theta_i$, i=1, ..., k where

$$s_i^t s_i = \sum_{j=1}^{k} s_i^2(j) = 1.$$

2) Select those pairs (θ_i, s_i) with the property $| s_i(k) | < 10^{-6}$.

3) For the values of i found in 2) recall the Lanczos vectors and compute

$$y_i = q_1 s_i(1) + q_2 s_i(2) + \cdots + q_k s_i(k).$$

The pairs (θ_i, y_i) are approximate eigenvectors with the property

$$\| By_i - y_i \theta_i \|_M = 10^{-6} \beta_{k+1}.$$

The θ_i will be accurate to about 10 significant figures. More precisely there is an eigenvalue λ of B such that

$$| \lambda - \theta_i | \leq 10^{-10} \beta_{k+1}^2 / gap$$

where gap $=$ min $| \mu - \theta_i | \approx$ min $\{ | \theta_{i+1} - \theta_i |, | \theta_i - \theta_{i-1} | \}$ and μ is the closest eigenvalue of B to θ_i except for λ. See [56].

The description given above ignored the important issue of detecting k, the right step at which to stop. See [26], [56] and [69].

Orthogonallty Loss

The preceding section may seem too good to be true. And it is. Finite precision arithmetic prevents the Lanczos vectors from being M-orthogonal or even close to it. In *Fig.1* we show that matrix $Q_k^t M Q_k$, where $Q_k = (q_1, ..., q_k)$. It should be the identity matrix.

This defect gave the Lanczos algorithm a bad reputation and only a few engineers persevered with it during the 1960s. See [77] & [78]. In 1971 C. C. Paige, in a remarkable thesis, was the first to explain this orthogonality loss in a satisfactory way. Once this understanding spread it was not difficult to find ways to handle the problem. In particular he showed that the original algorithm is not unstable; you can extract all eigenvalues.

It is beyond the scope of this brief paper to expound Paige's contribution. For a text book account for numerical analysts see [59, Chap.13] and for a more detailed, but lucid, presentation see two papers by Paige, [51] and [52].

What we can say is that there are three different ways of responding to orthogonality loss when executing the three term recurrence. The first, advocated by Paige himself (though not in the context of Mechanics), has been fully developed by Cullum and Willoughby, see [13], [14], and also Parlett & Reid [55]. The idea is to keep the original algorithm. The result is that the method becomes truly iterative, it does not stop at step n. Nevertheless it is possible to deduce the true eigenvalues of B from the computed tridiagonal T. However the number of steps required grows by a factor of 2 or 3 at least. Moreover eigenvectors must be computed separately. These programs seem best suited for computing <u>all</u> the eigenvalues of B. Engineering development along these lines are [18] and [77].

The opposite extreme explicitly applies the Gram-Schmidt process to make Bq_j orthogonal to q_1, q_2, ..., q_{j-2} at each step. Besides the cost of this extended computation there is the need to access all Lanczos vectors at each step. However the minimal number of steps will be taken. For short Lanczos runs of about 20 steps the cost of this approach is not heavy. See [10], [19], [20], [21], [28], [49], [50], and [75].

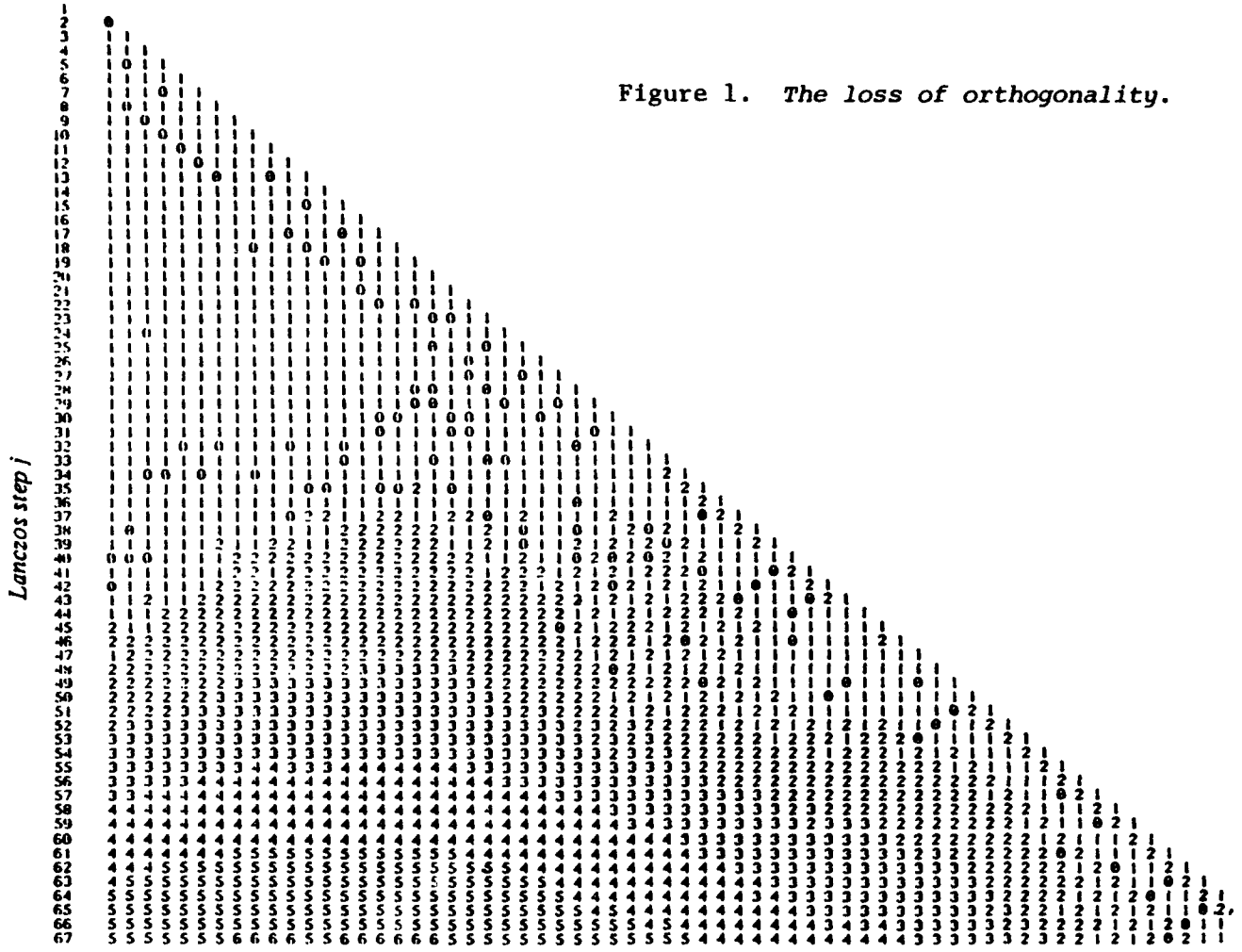


Figure 1. *The loss of orthogonality.*

Legend:

The (i, j) element indicates the inner product of the ith and jth Lanczos vectors computed by the three term reference.

In the matrix of *Fig.1* an integer m stands for the quantity $10^m\varepsilon$ where ε is the roundoff unit.

A third approach is based on the observation that it suffices to maintain semi-orthogonality to enjoy the main advantages of full-orthogonality. If the unit roundoff in the arithmetic unit is ϵ then two vectors of norm one are semi-orthogonal if their inner product does not exceed $\sqrt{\epsilon}$ in magnitude. Moreover semi-orthogonality among the Lanczos vectors can be maintained for about 1/3 the cost of maintaining ϵ-orthogonality. In particular the minimal number of Lanczos steps will be taken. See [58], [69] and [71].

It seems likely that in Mechanics it will be preferable to keep the number of calls on the operator B to a minimum. For applications in which the whole spectrum is wanted the first approach is the most attractive.

To complicate the issue there is the extra possibility of using block versions of the Lanczos algorithm. That leads to the next section.

Block Lanczos

Just as the power method is readily extended to a block form (namely subspace iteration) so is the Lanczos recurrence extended to work with several vectors simultaneously. A block size l is chosen and a sequence of $n \times l$ Lanczos matrices $Q_1, Q_2, ..., Q_k$ is constructed to satisfy

1. $Q_j^t M Q_j = I_l$

2. $BQ_j = Q_{j-1}B_j^t + Q_j A_j + Q_{j+1}B_{j+1}$

Here B_l is an $l \times l$ upper triangular matrix and A_l is $l \times l$ symmetric. The associated matrix T_k is no longer tridiagonal but has semi-bandwidth l. Consequently all calculations involving T_k are more costly than with simple Lanczos.

The original motivation for block versions of Lanczos was to allow the detection of multiple eigenvalues of B. In exact arithmetic the simple Lanczos algorithm cannot distinguish multiplicities but, in practice, roundoff error ensures that the right number of copies of an eigenvalue will be found, but at the cost of a few extra Lanczos steps, provided that the Lanczos vectors are kept orthogonal or semi-orthogonal. See [60].

Theoretically the simple Lanczos algorithm has the best approximation properties for a given number of calls on the operator B.

However the increasing complexity of modern operating systems and storage management has provided a powerful incentive to use block codes despite their increased cost per step. If the triangular factor L in $LDL^t = K - \sigma M$ cannot be stored in the fast memory then considerable transfer between primary and secondary storage is needed to invoke the operator B. In such cases it is advantageous to let B operate on several vectors at once. The precise number is problem and computer system dependent.

We can say that in each case there is a maximal number l such that the cost of applying B to l vectors is within 10% of the cost of applying B to one vector. In a good number of cases $l = 1$ but, in general l, as defined, is the correct block size - assuming that l n-vectors can be held in primary storage along with sections of L.

The block Lanczos program embedded in MSC/NASTRAN Version 65 permits the user to specify the block size. See [26],[39]. Block Lanczos methods are described in [25], [26], [28], [45], [63], [67] & [69].

8. LANCZOS VERSUS SUBSPACE ITERATION (SI)

A comparison between two implementations is presented in [46] although the experiments were made in 1980. Our only interest here is to illustrate the approximation power that comes from using combinations of the power vectors instead of using just the most recent one.

A three dimensional building frame was analyzed by the finite element method. There were 468 d.o.f. and the 60 dominant mode shapes were to be computed. This is more than 10% of the spectrum. The interesting question is how many calls on the operator subroutine [i.e. solve ($K - \sigma M$)$r = Mq$ for r] are needed for the computation of these 60 eigenvectors.

Recall that SI has to choose the subspace size. A well-known rule selects 68 but for the given application this is inefficient. We show below in Table 1 the number of operator calls required by several programs.

Table I†

	q	No. of steps	Matrix–vector ops.
SI(basic)	68	47	3196
SI(accelerated)	68	36	2448
SI(accelerated)	20	80	1600
LANSO	1	120	120

† q = dimension of subspaces used.

Several points need to be made in order to see the significance of Table 1.

i) The program SI (accelerated) incorporates a lot of clever tricks for improving the performance of SI. See [8] for the details. Consequently it is more instructive to consider a straight forward implementation, namely SI (basic), to appreciate the theoretical improvement that comes from using a Krylov subspace. Figures are not available for SI (basic) used with $q = 20$ but we may infer that at least 2000 calls on the operator subroutine would be made. Yet the bottom line of Table 1 shows that only 120 are needed by a rival method starting from a random vector.

ii) SI (accelerated) used 12 different shifts (i.e. factorizations of $K - \sigma M$) to achieve its improvements whereas LANSO used only one. However LANSO could also benefit from use of more than one shift as is shown in [20]. The benefit is in storage requirements, not calls on the operator. For example, one could use 4 shifts in turn making about 31 or 32 calls on the operator each time while picking up 15 eigenvectors. In this way there only needs to be storage for 35 vectors. That means that larger problems could be solved completely inside the fast memory. The selection of shifts is an interesting topic. See [19] and [35].

iii) There is a price to be paid for achieving this very small number of calls on the operator subroutine. It is necessary to keep all the Lanczos vectors somewhere because they are recalled at the end to accumulate the eigenvectors. It turns out that they are needed more often than that because of roundoff error. In the simplest implementations the new Lanczos vector is explicitly orthogonalized against all previous Lanczos vectors at each step. This is overkill. Nevertheless it is necessary to do the orthogonalization from time to time about (one step in four, on the average) in order to keep the number of steps to a minimum.

More information on these topics is contained in [19], [21], [48].

iv) It would be more informative to compare SI with a block Lanczos code but we are not aware of such a trial.

9. SINGULAR M

The null space of M is also the nullspace of $B = (K - \sigma M)^{-1} M$ (i.e. $Mv = 0$ if, and only if, $Bv = 0$). Every nontrivial vector in this subspace $N(B)$ may be considered an eigenvector of B with eigenvalue ∞. Complementary to $N(B)$ is $R(B)$, the range space of B. It is spanned by the eigenvectors belonging to the finite eigenvalues.

By restricting attention to $R(B)$ the regular case is recovered. Of course $R(B)$ is not known but in exact arithmetic it is easy for all computed vectors to be kept in $R(B)$. It suffices that the starting vectors, for SI and Lanczos, be in $R(B)$. Unfortunately rounding errors introduce into all computed vectors components in $N(B)$. With Lanczos these components grow steadily as the computation proceeds but this feature is not at all evident because the M-inner product is blind to such errors. Engineers are often tempted to stay in $R(B)$ by using static condensation, see [32], but this usually spoils the sparsity structure of the reduced matrices.

There is an easy cure. Each approximate eigenvector y may be projected back onto $R(B)$ by forming By and then normalizing. However this is expensive. If m approximate eigenvectors are computed using only $2m$ calls on B then a further m calls on B to purify the eigenvectors adds approximately 50% to the cost.

Is there a way to purify y without making any more calls on B ?

With the Lanczos algorithm there is a nice trick. See [48]. It is easy to explain and implement.

Let (θ_i, y_i) be an approximate eigenpair with

$$y_i = Q_j s_i = \sum_{k=1}^{j} q_k s_i(k).$$

See section **7** for notation. The next, so-far-unused Lanczos vector q_{j+1} is available. It turns out that

$$By_i = y_i\theta_i + q_{j+1}\beta_{j+1}s_i(j)$$

to within roundoff terms. Consequently it is only necessary to replace y_i by

$$\bar{y}_i := y_i + q_{j+1}(\beta_{j+1}s_i(j)/\theta_i).$$

The quantity $\beta_{j+1}s_i(j)$ will already be known since it provides an error bound on y_i. There is no need to have $s_i(j)^2 <$ unit roundoff (from section **7**) so the correction to y_i is far from negligible. See the Table in [48] for the striking effect.

This modification makes a small improvement even when M is invertible. Consequently it should be made automatically and there is no need for the program to know whether or not M is singular. A beautiful result.

10. LOOKING AHEAD

The linear eigenvalue problem $K - \lambda M$, with real eigenvalues, is in good shape but there are many challenges ahead.

1. Efficient, robust techniques for quadratic eigenvalue problems (as indicated in Section **3**). Standard practice encourages reduction to a linear problem, see [23] and [36] for example, but this may not be optimal. See [66] and [74].
2. Problems with complex conjugate pairs of eigenvalues (as indicated in Section **3**). The matrices described in this paper can be extended to the nonsymmetric case. See [30], [72] for SI and [24], [54] for Lanczos. Also see [64].
3. Exploiting vector computers to the full.
4. The effect of parallel computers.

That can wait for the next issue of State-of-the-Art!

REFERENCES

1. "Matrix Eigensystem Routines - EISPACK Guide (1976)," in *Lecture Notes in Computer Science, Second Edition*, ed. C. B. Moler, vol. 6, Springer, New York.

2. "Matrix Eigensystem Routines - EISPACK Guide Extension (1977)," in *Lecture Notes in Computer Science, Second Edition*, ed. C. B. Moler, vol. 51, Springer, New York.

3. *NASTRAN Users Manual (1977)*, NASA Langley Research Center, Hampton, Virginia.

4. *HARWELL Subroutine Library (1981)*, Computer Science and System Division, Atomic Energy Research Establishment, Harwell, Oxfordshire, England.

5. Aasleend, L. and P. Bjorstad (1983), "The generalized eigenvalue problem in ship design and offshore industry," in *Matrix Pencils*, ed. A. Ruhe.

6. Barth, W., R. S. Martin, and J. H. Wilkinson (1967), "Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection," *Numer. Math.*, vol. 9, pp. 386-393.

7. Bathe, K.-J. and E. L. Wilson (1976), *Numerical Methods in Finite Element Analysis*, Prentice, Englewood Cliffs, New Jersey.

8. Bathe, K.-J. and S. Ramaswamy (1980), "An accelerated subspace iteration method," *Comput. Methods Appl. Mech. Engrg.*, vol. 23, pp. 313-331.

9. Butscher, W. and W. E. Kammer (1976), "Modification of Davidson's method for the calculation of eigenvalues and eigenvectors of large real symmetric matrices: " Root- homing procedure"," *J.Comput. Phys.*, vol. 20, pp. 313-325.

10. Carnoy, E. and M. Geradin (1982), "On the practical use of the Lanczos algorithm in finite element applications to vibration and bifurcation problems," in *Proceedings of the Conference on Matrix Pencils, Held at Lulea, Sweden, March 1982, University of Umea*, ed. Axel Ruhe, pp. 156-176, Springer Verlag, New York.

11. Clint, M. and A. Jennings (1970), "The evaluation of eigenvalues and eigenvectors of real symmetric matrices by simultaneous iterations," *Compt. J.*, vol. 13, pp. 76-80.

12. Corr, R. B. and A. Jennings (1976), "A simultaneous iteration algorithm for symmetric eigenvalues problems," *Internat. J. Numer. Methods Engrg.*, vol. 10, pp. 647-663.

13. Cullum, J. and R. A. Willoughby (1979), "Lanczos and the computation in specified intervals of the spectrum of large sparse real symmetric matrices," in *Sparse Matrices and their uses*, ed. Duff, pp. 220-255, Acad. Press.

14. Cullum, J. and R. A. Willoughby (1984), "Lanczos Algorithms for Large Symmetric Matrices," *Users Guide*, vol. 1 and 2, Birkhauser, Boston.

15. Cuppen, J. M. M. (1981), "A Divide and Conquer Method for the Symmetric Tridiagonal Eigenproblem," *Num. Math.*, vol. 36, pp. 177-195.

16. Daniel, W. J. T. (1980), "Modal Methods in Finite Element Fluid - Structure Eigenvalue Problems," *Int. J. Num. Meth. Eng.*, vol. 15, no. 8.

17. Dongarra, J. and D. Sorenson (1987), "A Fully Parallel Algorithm for the Symmetric Eigenvalue Problem," *SISSC.* To appear

18. Edwards, J. T., D. C. Licciardello, and D. J. Thouless (1979), "Use of the lanczos method for finding complete sets of eigenvalues of large sparse symmetric matrices," *J. Inst. Math. Appl.*, pp. 277-283.

19. Ericsson, T. and A. Ruhe (1980), "The spectral transformation Lanczos method for the numerical solution of large sparse generalized symmetric problems," *Math. Comp.*, vol. 35, pp. 1251-1268.

20. Ericsson, T. and A. Ruhe (1982), "STLM - a software package for the spectral transformation Lanczos algorithm," UMINF-101.82, U. Umea.

21. Ericsson, T. (1983b), "Algorithm for large sparse symmetric generalized eigenvalue problem," UMINF-108.83, U. Umea.

22. Felippa, C. A. (1977), "Procedures for Computer Analysis of large nonlinear structural systems," Proceedings of the International Symposium on Large Engineering Systems *held at the Univ. of Manitoba, Winnipeg, Canada in August 1976.*, Pergamon Press.

23. Fricker, A. J. (1983), "A Method for Solving High - Order Real Symmetric Eigenvalue Problems," *Int. J. Num. Meth. Eng.*, vol. 19, no. 8.

24. Gerardin, M. (1977), "Application of the Biorthogonal Lanczos Algorithm," *LTAS.*

25. Golub, G. H. and R. Underwood (1977), "The block Lanczos method for computing eigenvalues," in *Mathematics Software III*, ed. J. R. Rice, pp. 361-377, Academic Press, New York.

26. Grimes, R., J. G. Lewis, and H. Simon (1986), "The Implementation of a block, shifted and inverted, Lanczos algorithm for eigenvalue problems in structural engineering," Tech. Report ETA-TR-39, Boeing Computer Services, Seattle, WA..

27. Gupta, K. K. (1974), "Eigenproblem solution of damped structural systems," *Int. J. Num. Meth. Eng.*, vol. 8, pp. 877-911.

28. Gupta, K. K. (1978b), "Development of a finite dynamic element for three vibration analysis of two-dimensional structures," *Internat. J. Numer. Methods Eng.*, vol. 12, pp. 1311-1327.

29. Iyer, M. S. (1981), "Eigensolution using Langrangian Interpolation," *Int. J. Num. Meth. Eng.*, vol. 17, no. 10.

30. Jennings, A. and W. J. Stewart (1975), "Simultaneous iteration for partial eigensolution of real matrices," *J. Inst. Maths. Applics.*, vol. 15, pp. 351-361.

31. Jennings, A. and T. J. A. Agar (1981), "Progressive Simultaneous Inverse Iteration for Symmetric Linearized Eigenvalue Problems," *Computers and Structures*, vol. 14, no. 1-2.

32. Jennings, A. (1973), "Mass condensation and simultaneous iteration for vibration problems," *Int. J. for Num. Methods in Engng.,*, vol. 6, pp. 543-552.

33. Jennings, A. (1977), in *Matrix Computation for Engineers and Scientists*, Wiley, New York.

34. Jennings, A. (1981), "Eigenvalue methods and the analysis of structural vibration," in *Sparse Matrices and their Uses*, ed. Duff, pp. 109-138.

35. Jensen, P. S. (1972), "The solution of large eigenproblems by sectioning," *SIAM J. Numer. Anal.*, vol. 9, pp. 534-545.

36. Johnson, C. P. etc (1980), "Quadratic Reduction for the Eigenproblem," *Int. J. Num. Meth. Eng.*, vol. 15, no. 6.

37. Kats, J. M. van and H. A. van der Vorst (1977), "Automatic monitoring of Lanczos schemes for symmetric or skew symmetric generalized eigenvalue problems," Acad. Comp. Centrum Report TR-7, U. Utrecht.

38. Komzsik, L. and G. A. Dilley (1985), *Practical Experiences with the Lanczos Method.* private communication concerning MSC/NASTRAN Version 65

39. Lanczos, C. (1950), "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Nat. Bur. Stand.*, vol. 45, pp. 255-282.

40. Levit, Itzhak (1984), "A New Numerical Procedure for Symmetric Eigenvalue Problems," *Computers and Structures*, vol. 18, no. 6.

41. Lewis, J. G. and R. G. Grimes (1981), "Practical Lanczos algorithms for solving structural engineering problems," in *Sparse Matrices and their Uses*, ed. Duff, pp. 349-355, Acad. Press.

42. Loh, C. H. (1984), "Chebyshev Filtering and Lanczos' Process in the Subspace Iterative Method," *Int. J. Num. Meth. Eng.*, vol. 20, no. 1.

43. Martin, R. S. and J. H. Wilkinson (1968), "Reduction of the symmetric eigenproblem $Ax=\lambda Bx$ and related problems to standard form," *Numer. Math.*, vol. 11, pp. 99-110.

44. Matthies, H. and G. Strang (1979), "The Solution of Nonlinear Finite Element Equations," *Int. J. Num. Meth. Eng.*, vol. 14, pp. 1613-1626.

45. Matthies, H. G. (1985), "A Subspace Lanczos Method for the Generalized Symmetric Eigenproblem," *Computers and Structures*, vol. 21, pp. 319-325.

46. Nour-Omid, B., B. N. Parlett, and R. L. Taylor (1983), "Lanczos versus subspace iteration for solution of eigenvalue problems," *Internat. J. Numer. Methods Engrg.*, vol. 19, pp. 859-871.

47. Nour-Omid, B., B. N. Parlett, and R. Taylor (1983), "Lanczos versus Subspace Iteration for Solution of Eigenvalue Problems," *Int. J. Num. Meth. Eng.*, vol. 19, no. 6.

48. Nour-Omid, B., B. N. Parlett, T. Ericsson, and P. S. Jensen (1987), "How to implement the Spectral Transformation," *Math. Comp.*, vol. 48.

49. Ojalvo, I. U. and M. Newman (1970), "Vibration modes of large structures by an automatic matrix-reduction method," *AIAA Journal*, vol. 8, pp. 1234-1239.

50. Ojalvo, I. U. (1985), "Proper Use of Lanczos Vectors for Large Eigenvalue Problems," *Computers and Structures*, vol. 20, no. 1-3.

51. Paige, C. C. (1976), "Error analysis of the Lanczos algorithms for tridiagonalizing a symmetric matrix," *J. Inst. Math. Appl.*, vol. 18, pp. 341-349.

52. Paige, C. C. (1980), "Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem," *Linear Algebra Appl.*, vol. 34, pp. 235-258.

53. Papadrakasis, M. (1984), "Solution of the Partial Eigenproblem by Iterative Method," *Int. J. Num. Meth. Eng.*, vol. 20, no. 12.

54. Parlett, B. N., D. R. Taylor, and Z. A. Liu (1985), "A Look-Ahead Lanczos Algorithm for Unsymmetric Matrices," *Math. Comp.*, vol. 44, pp. 105-124.

55. Parlett, B. N. and D. S. Scott (1979), "The Lanczos algorithm with selective reorthogonalization," *Math. Comp.*, vol. 33, pp. 217-238.

56. Parlett, B. N. and J. K. Reid (1981), "Tracking the progress of the Lanczos algorithm for large symmetric matrices," *IMA J. Numer. Anal.*, vol. 1, pp. 135-155.

57. Parlett, B. N. and B. Nour-Omid (1985), "The Use of Refined Error Bounds when Updating Eigenvalues of Tridiagonals," *J. Lin. Alg. & Appls.*, vol. 68, pp. 179-220.

58. Parlett, B. N., B. Nour-Omid, and J. Natvig, "Implementation of Lanczos algorithms on Vector Computers," *Supercomputer Applications*, Plenum Press, New York, 1985.

59. Parlett, B. N. (1980), in *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ.

60. Parlett, B. N. (1980b), "How to solve $(K-\lambda M)z = 0$ for large K and M.," *Numer. Methods Engrg.*, Dunod, Paris.

61. Parlett, B. N. (1984), "The Software Scene in the Extraction of Eigenvalues from Sparse Matrices," *SIAM J. Sci. Stat. Comput.*, vol. 5, pp. 590-603.

62. Robati, P. (1984), "The Accelerated Power Method," *Int. J. Num. Meth. Eng.*, vol. 20, no. 7.

63. Ruhe, A. (1979), "Implementation aspects of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices," *Math. Comp.*, vol. 33, pp. 680-687.

64. Saad, Y. (1981), "Krylov Subspace methods for solving large unsymmetric linear systems," *Math. Comp.*, vol. 37, pp. 105-126.

65. Schwarz, H. R. (1977), "Two algorithms for treating $Ax=\lambda Bx$," *Comput. Methods Appl. Mech. Engrg.*, vol. 12, pp. 181-199.

66. Scott, D. S. and R. C. Ward (1982), "Solving symmetric-indefinite quadratic lambda-matrix problems without factorization," *SIAM J. Sci. Statist. Comput.*, vol. 3, pp. 58-67.

67. Scott, D. S. (1979), "Block Lanczos software for symmetric eigenvalue problems," ORNL Report CSD-48 , Oak Ridge, Tennessee.

68. Scott, D. S. (1980), "The advantages of inverted operators in Rayleigh-Ritz approximations," *SIAM J. Sci. Statist. Comput.*, vol. 3, pp. 68-75.

69. Scott, D. S. (1981), "The Lanczos algorithm," in *Sparse Matrices and Their Uses*, ed. I. S. Duff, pp. 139-159, Academic Press, New York.

70. Scott, D. S. (1981b), "Solving sparse symmetric generalized eigenvalue problem without factorization," *SIAM J. Numer. Anal.*, vol. 18, pp. 102-110.

71. Simon, H. D. (1984), "The Lanczos algorithm with partial reorthogonalization," *Math. Comp.*, vol. 42, pp. 115-142.

72. Stewart, W. J. and A. Jennings (1981), "A simultaneous iteration algorithm for real matrices," *ACM TOMS*, vol. 7, pp. 184-198.

73. Utku, Senol etc (1984), "Computation of Eigenpairs of $Ax=\lambda Bx$ for Vibration of Spinning Deformable Bodies," *Computers and Structures*, vol. 19, no. 5/6.

74. Veselic, K. (1980), "On Optimal Linearizations of a Quadratic Eigenvalue Problem," *Linear & Multilinear Algebra*, vol. 8, pp. 253-258.

75. Weingarter, V. I. etc (1983), "Lanczos Eigenvalue Algorithm for Large Structures on a minicomputer," *Computers and Structures*, vol. 16, no. 1-4.

76. Wellford, L. Carter etc (1980), "Post-buckling Behaviour of Structures using a Finite Element - Nonlinear Eigenvalue Technique," *Int. J. Num. Meth. Eng.*, vol. 15, no. 7.

77. Whitehead, R. R., A. Watt, B. J. Cole, and I. Morrison (1977), "Computational methods for nuclear shell-model calculations," *Advances in Nuclear Physics*, vol. 9, pp. 123-176, Plenum Press, New York.

78. Whitehead, R. R. (1972), "A numerical approach to nuclear shell-model calculations," *Nucl. Phys.*, vol. A182, pp. 290-300.

79. Wilkinson, J. H. and C. Reinsch (1971), *Handbook for Automatic Computation: vol.II, Linear Algebra* , Springer-Verlag, New York.

80. Wilkinson, J. H. (1966), *The Algebraic Eigenvalue Problem*, Oxford Univ. Press.

81. Wilson, E. L., M. W. Yuan, and J. M. Dickens (1982), "Dynamic analysis by direct superposition of Ritz vectors," *E.E.Str. Dyn.*, vol. 10, pp. 813-821.

82. Yang, W. H. (1983), "A method for Eigenvalues of Sparse λ- matrices," *Int. J. Num. Meth. Eng.*, vol. 19, no. 6.

END

Filmed

5-89

DTIC